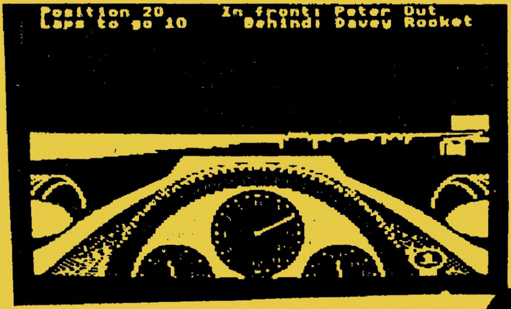
ISSN 0112-3351

BEEBLET

Its Here! 2
Auckland Report 4
Comms User Group 4
BASIC ROM User Guide - Review 5
Revs - Review 7
Deals and Discounts 10
Guide to the BBC Roms - Review 11
Acorn Electron Advanced User Guide - Review 13
Classified Ads 14
Electron Column 15
Mailbag 18
Beebs in the Backwoods 22
Simple Database II 24
Billboard? 31
Title Pages in Mode 7 33
Civilised Rounding 39
Local Groups 40

VOLUME 3 NUMBER 10 OCTOBER 1985



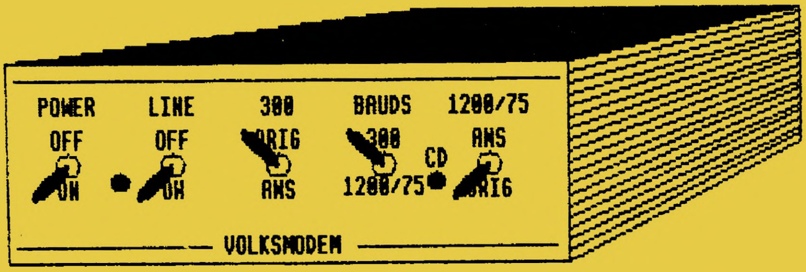
Position 20 In front: Peter Out
Laps to go 10 Behind: Davey Rocket

BBC / ACORN

COMPUTER USER GROUP NZ (inc)

Announcing the ...

VolksModem



IT'S HERE!

After months of planning and development, the User Group's very own modem - the VolksModem - is here.

The prototype was submitted to the Post Office in early November and we confidently expect them to give it type approval for direct connection to the telephone network.

SPECIFICATIONS

The VolksModem is powered by a 230V to 12V AC transformer (supplied) which plugs into the back. The modem is turned on with the leftmost switch on the front panel - lighting up a red LED just to its right.

Also supplied is a cable connecting the modem to the BBC microcomputer. This has a 25 pin plug on one end to connect to the RS232 connector on the back of the modem, and a 5 pin connector to plug into the BBC's RS423 socket.

To use the modem you should have one of the new push-button "Pert" phones, since these have a BT plug which plugs into a BT socket in the wall. As Pert phones are the new standard, the two monthly rental on them is now the same as for the old phones.

According to the Post Office it will cost you $37 to change your rotary dial phone to a Pert. Note that this is cheaper than the $45 to add a single BT socket without doing anything to the phones. However, you may want to pay another $45 to add another socket near your computer if the phone is a fair distance away.

We will supply a telephone extension cord - with a male BT plug on each end. One end plugs into the LINE socket on the back of the VolksModem, and the other end into the BT wall socket. Then you plug your phone into the PHONE socket on the back of the modem. (Your phone will fit nicely on top of the Volksmodem). The standard length of the supplied telephone extension cord will be 3 metres. If you want a longer cord, please say so when you order (extra cost is around $1 per metre).

So long as the LINE switch on the front panel is in the OFF position, the phone can be used normally. Once you have used the phone to call the number you want to communicate with, switching down to the ON position (as shown) will "hold" the line and the phone should be hung up - the connection will not be lost. The green Carrier Detect (CD) LED on the right of the front panel will light up when the remote modem talks to you.

The right hand three switches on the front panel set the VolksModem mode. When the BAUDS switch is up (as in the picture), the switch to its left controls whether you are in 300/300 bps ORIGinate or ANSwer mode, and the position of the right hand switch is irrelevant. If the BAUDS switch is down, the right hand switch controls whether you are transmitting at 1200 bps and receiving at 75 bps (ANSwer), or receiving at 1200 bps and transmitting at 75 bps (ORIGinate). The latter mode is what would be used for calling Videotex services.

SOFTWARE

It is probable that some software will be supplied with the VolksModem, allowing bulletin board access and file transfer using the Xmodem protocol (see my article in the April 1985 "Beeblet"), but probably not PRESTEL protocols as used by the Videotex services. It depends on how much time I get to work on the program in the next few weeks.

So far as commercial programs go, I can recommend the COMMSTAR package available from Britain for 27 pounds. This does just about everything you need, including Xmodem file transfer and PRESTEL, but you'll need a spare ROM socket.

PRICE AND ORDERING

We have arranged for the modems to be manufactured for us. The price for User Group members looks like being around $210 all up. Because of labour costs this is higher than expected earlier, but is still an extremely good deal compared with what is available commercially. With Post Office approval you can be assured of getting a quality product.

To start making the modems we need orders, with a $50 deposit, right now. The balance will be payable on delivery.

The sooner we know how many we will be building, the sooner we will be able to order parts and get cracking. We hope to get the first ones out by Christmas, but cannot promise that.

To order, send $50 and a piece of paper detailing:

your name, address and telephone number
your User Group membership number
how long a telephone cable you want (standard is 3 metres)

to P.O.Box 9592, Wellington.

Orders will be filled on a first come first served basis. When we know the final cost, we will contact you and ask for the balance, then send out your VolksModem.

Neil Gordon

THE AUCKLAND REPORT

Steve Williams

Last meeting was held as a number of small groups, and this appeared to be a popular format. Most popular topics for the groups were Comms and Business software. Meetings will continue in this format with subjects for groups being chosen "on the night".

COMMS USER GROUP

Anyone with a particular interest in Comms (or just want to know more about it) please write to COMMS INTEREST GROUP, PO BOX 6662, WELLESLEY STREET, AUCKLAND. This group has been formed to bring people with a common interest together, and is not confined to just BBC owners. Please send an SAE for a copy of ONLINE, our newsletter.

BASIC ROM USER GUIDE

by Mark Plumbley

Reviewed by Steven R.

359 pages, published by Adder Publishing

Review copy supplied by Whitcoulls, price $35.95.

This book is for both the BBC and the ELECTRON, since both have the same BASIC.

As the BASIC ROM is in machine code, anybody wanting to look at it is going to need a passing familiarity with assembly language. This is not stated, but as with the GUIDE TO THE BBC ROMS, is implied in the description on the back cover.

Having said that, you can get a lot out of this book whether you know assembler or not, because each routine is described in English, although using a few assembly terms. Also, assembly language and the 6502 are described in the first 21 pages of the book, including a list of the 6502 instructions and their functions, which will give you a good start.

Unlike the GUIDE TO THE BBC ROMS, this book deals only with BASIC. It covers BASIC 1 and 2, and includes helpful hints, like how to recover from errors, extend BASIC by error trapping, change modes from inside FNs or PROCs, overlay PROCs or FNs to let you have very large programs, list active variables, salvage bad programs...

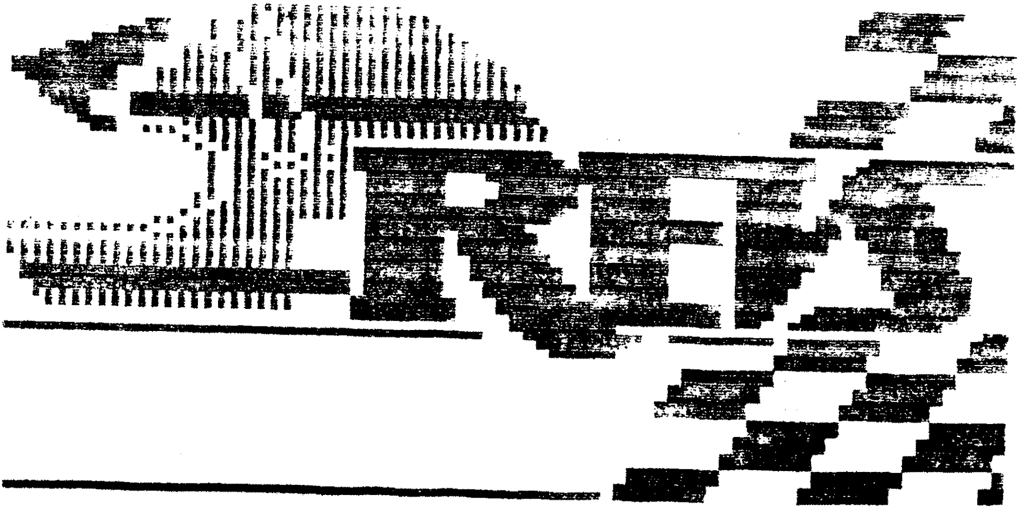
To help you to follow the machine code routines, a machine code dissassembler is included, which will dissassemble one instruction per CALL. This means building a program around it to get a complete listing, but a simple REPEAT:UNTIL loop may well suffice.

The book contains a chapter on each area of BASIC, including program storage, memory use, expression evaluation, program control mechanisms (Eg. GOTO, REPEAT UNTIL, ON ERROR), the assembler, and how BASIC does the statement handling, manages the stacks and variables, and so on. There are also chapters on extending BASIC, as mentioned above. Each chapter contains a functional description of the major routines in that area of BASIC, with comments such as potential improvements, bugs and possible problems, program listings to demonstrate using a routine or to help to decipher it, and the occasional algorithm or two. There is usually no listing of the ROM routines involved.

There are seven appendices, covering syntax definitions, ROM summary by address, the 6502 instruction set, ROM summary by Keyword, Operating System calls and vectors, OSBYTE/*FX call summary, and variable locations. There are no routines that are obviously missing from the ROM summaries.

In addition to the contents and appendices, there are a bibliography, a glossary of terms, and an index. The index is reasonable, but you may need to try more than once to get to where you want to go.

To summarise, this is a book for people who want to know more about the workings of the BASIC ROM, and who are not afraid of assembly language. Although usually no listings are given, all entry points are given in both address and keyword order, and a disassembler is supplied. Some routines are also given to extend BASIC, such as a CLI for BASIC keywords so that you can add your own.
This seems to be a comprehensive, friendly book, easy to read and use. In combination with the GUIDE TO THE BBC ROMS, it forms a reasonably complete guide to the ROM software supplied with the BBC or ELECTRON.



REVS

Reviewed by Keith F

Acornsoft's "Revs" $56.95 disc, also tape available (no difference).
Can be played with joystick (difficult), or with keyboard (virtually impossible.)

I shall start off by saying that this is not your usual "arcade" game. There is no easy version to warm you up. No, this is more of a simulation than a game. The player takes the role of a formula 3 racing car driver. He or she sees exactly what a real driver would see - ie you are really in the car looking out.

Our player has the following equipment: a fully functional rev counter, a gearshift (5 forward, neutral, reverse), accelerator, brake, and steering. In addition our hero(ine - typist) also has front and back wings.

These can both be set to 40 different settings. Different combinations do different things. For instance, if a beginner takes the book's advice for setting up the car, the result will be a good handling car with lots of grip on the curves, but a poor top speed. By lowering the wing settings, you get more speed, but the corners are more difficult to negotiate. (The beginner may not at first believe this to be possible!). However, we also have a new problem as well at this point. Not only are the curves more difficult to take, but the car will no longer handle properly, unless you are very lucky.

You have to get the front and back wings to balance, otherwise you will get one of two problems: understeer, or oversteer. Understeer is where you turn the front wheels but the car doesn't want to know. This usually results in going head first off the circuit and initially, the driver wondering what he did wrong. Oversteer is where you turn the front wheels into the corner and the back wheels get jealous. They demonstrate this by taking the rear of the car around the outside of the corner in an attempt to overtake the front. Until one becomes skilled, the rear wheels usually win and you will find scenery going in all directions.

This is the only point at which the game is not true to life - not through a flaw of the program - but simply because the computer cannot simulate the G-forces a real driver would feel. The only clues you get that the car is sliding is that the tires start to "squeal" and you can "see" that the car is not doing what it should. It takes a lot of practice to pick up the signals before it is too late and you go off.

So now, having blinded you with science, here is what it is like to play. There are two modes: testing, and racing.

In testing you are alone on the track, an excellent learning tool. You are sitting in the middle of the track. First start your car, select first gear and drive off. Easy huh? Wrong!! Starting is easy. Press "T" until the engine fires. Actually getting forward motion is more tricky. For a start, the useable rev band is between 5 and 6 thousand. Anything else is useless. Also (using a joystick), changing gear properly takes some mastering. I'm not going to explain, you can read the book!

My advice to the learner is to master starting off and changing gear before you worry about cornering style. This you will find easier than you might think, for when you crash, your car is put back at the start. Few people negotiate the first curve in the first half hour. (If you do and are feeling smug, just remember that it should be taken in fourth gear.)

The next step is to get right round the circuit. Study the book, start up and into 3rd gear. Stay there for a full lap. This will give a lap time of around 2 minutes. As your confidence builds, start to use 4th and 5th. Once you reach 1m 35 secs you are ready to enter a novice race. Do not deviate from the beginners wing settings until you have won a couple of novice races. Unless you are very good, you will learn bad habits and/or confuse yourself.

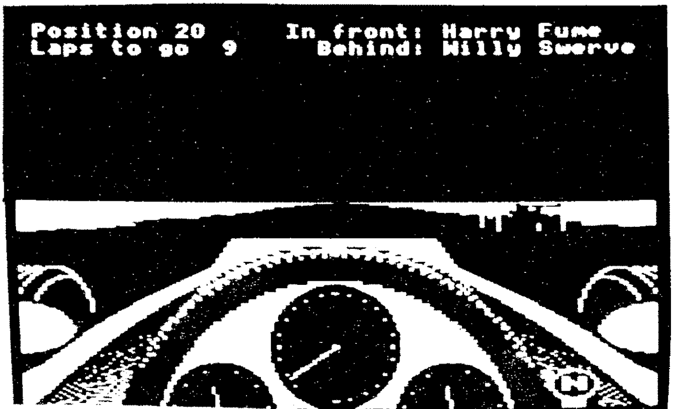
To enter a race you must enter official practice for one of the three classes:
1. Novice: for all those who can't break a lap time of 1:27 or those who can't do 5 laps without coming off.
2. Amateur: for those unable to break 1:26 but can do 5 laps around 1:27 without crashing.
2. Professional: must be sub 1:25 to stand a chance of victory, and an average lap time around 1:26

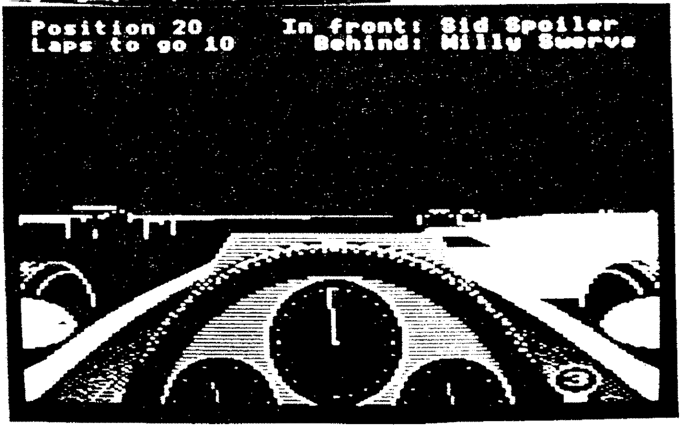
Note that the gap between amateur and professional looks small. It is not. It is immense. As an indication, David Hunt, the racing driver who assisted in development of Revs, has a best of 1:24.6 I myself have 1:25.8 but have heard rumours of a sub 1:23 lap.

Your practice session has other cars in it. They will knock you off the circuit, get in your way, and generally make life awkward. The best time you get decides where you start in a list of 20 cars.

Once racing, you will have a whole lot more to learn, such as that a car in front of you will disturb the air and stop your wings from working properly. But you can find that out for yourself! A full 20-lap race will result in at least half an hour of intense concentration which will leave you totally drained, even at novice level.
If you love racing, it's worth buying a BBC for this. If you want a 5 or 10 minute thrill, forget it. Buy Pole Position instead. My own opinion is that Revs is unsurpassed by any other racing game so far.

Other advantages are a multi-player option and the promise of more circuits to come. I myself am considering buying a BBC for the sole reason of playing this game!!





Confidential to Members. 11 November 1985

DEALS AND DISCOUNTS

We are currently holding stocks as follows:-

Disks (box of 10)	3M DSDD 5.25"	$55.00
"	3.5"	$85.00
Tapes(packet of 20)	Tracer C10	$30.00 + $2.00 pp
Tape(New Member)	EMI C30	$5.00
Disk storage box	CS (cap. 50)	$40.00
Paper fan-fold	M & P 1110/1	$5.00 per 200 sheets
Beeblets Vol 1	4,5,6,10,11	$0.50 each
" Vol 2	all except May	$1.50 each
" Vol 3	1,2	$1.50 each
Beeblet Covers	Pre-V-Dex(2 Vols)	$4.50
Cables	Printer	$25.00
Printer ribbons	Logitec	$10.00

Wellington members these are available at both monthly meetings and workshops.
For other members postage will be arranged on receipt of your cheque.

Bulk Deals will be arranged as follows when the required number of orders for the bulk deal have been received.

Printers	Solar 120EX	$575.00 + freight
	Solar 180EX	$760.00 + freight
	Solar 1800	$1030.00 + freight
Disk-drives	TEAC FD-55F	$385.00 + freight
" single kit	Drv+Box+Cbls	$455.00 + freight
" double kit	Drv(2)+Box+Cbls	$840.00 + freight
BBC DFS Kit		$275.00
Modems	Volksmodem	$210 approx $50.00 deposit
Monitors Green	BMC 12"	$200
" Colour	TVM 13"	$1050-1100 range

[40 x 400 lines(interlaced), 14MHz bandwidth 0.39mm pitch]

Initial orders are received without payment. When the required numbers to achieve a bulk order are received, you will be asked to confirm your order together with your cheque for payment. Goods will be freighted to out of Wellington members as soon as the goods are received. Please include your telephone number with your order.

For non-members there is an additional charge of $10.00 per order.

GUIDE TO THE BBC ROMS

by Don Thomasson

Reviewed by Steven R.

231 pages, published by Melbourne House. Review copy supplied by Whitcoulls, price $49.95.

Naturally, anybody wanting to know about the insides of the BBC ROMs is going to need to know assembly language (machine code). This is not stated, but is implied in the description on the back cover.

Quoting from the back cover, the purpose of this book is "to provide an 'at a glance' picture of the fundamental concepts of the BBC ROMs." To achieve this, the entry address of each ROM routine is listed, followed by a summary of each aspect of its function.

The introduction explains that because there are different versions of BASIC and the OS, the entry points have been given for the tape version of the 1.2 OS, and presumably BASIC 1, and the book attempts to provide enough guidance, for those with other versions, to "make exploration profitable". I say "presumably BASIC 1" because although I couldn't find any mention of the version of BASIC, the initialisation jump in the BASIC ROM is given as &801F, the execution address saved with BASIC 1 programs.

Having read the introduction, I wondered why a listing of each routine had not been provided, possibly with entry points for different versions of the ROMs. This would have meant that the comments could have been included in the dissassembly, and would have made reference to a routine easier. A dissassembler is provided in an appendix, but the reader must produce his own listing of the code for reference. The description usually includes the instruction being executed as well as giving an idea of the purpose, so the book is fairly readable without a listing to refer to, but occasionally I was left a little in the dark.

The BBC design segregates the functions of BASIC and the OS, unlike some micros which intertwine BASIC and the OS inseparably, and the BBC OS is really an interface. This means that apart from calculations, a language usually performs functions by doing something on the order of setting up a parameter block, and the OS does the rest. Because of this, the GUIDE TO THE BASIC ROMS is mainly concerned with the operating system, and in fact only spends one chapter on BASIC. This seems reasonable, as most routines required by a programmer will be in the OS.

The book contains detailed descriptions of all routines within the ROMs except speech. Because the author didn't have the speech facility, he was restricted to making informed guesses in places, and couldn't cover this as completely as other routines. That may be a drawback if you want to dabble in this area, although you have been given a good start.

As you will have gathered, the information presented is from the author's own analysis, which appears to be as complete as is possible from the examination of a BBC, without having the Acorn design documentation.

The GUIDE contains an analysis of the ROM routines, referenced by an address list at the end of the introduction, and by a quite detailed index at the back. It also has an appendix containing a three page disassembler listing, and a one and one half page program to sort a list of link addresses for you (equating link destination with the link address referred to by a jump or call), when you have found them.

Structured programmers will wince - GOTO and GOSUB are used throughout, without a single PROC or FN, although REPEAT UNTIL loops are used. Although there are some longer variable names, there are also plenty of meaningless two letter names which could have been longer. Just type it in and use it - following the spaghetti is not easy!

If you are prepared to print a dissassembly listing, or to read it from the screen in conjunction with the book, then this book contains enough information for you to use any routine in ROM, indexed sufficiently for you to be able to find the description quickly. It is self contained, and complements the U.G. and A.U.G.

If you want a book which supplies listings, then there are three other ROM books listed in the Technomatic adverts which may make it to NZ, and which may contain listings.

THE ACORN ELECTRON ADVANCED USER GUIDE

written by Mark Holmes & Adrian Dickens. Published by Adder Publishing, for Acornsoft Limited. Review copy supplied by Whitcoulls
price $35.95

Reviewed by Steve Cosgrove

Acorn have changed their documentation policy for the Electron. Much of the documentation is the same as that used for the BBC, but it is published in different books. The first thing I didn't like was the EUG index. It does not exist. The VIEW manual index I had to make myself. (See last month's Beeblet.) I was most relieved to find an index in the EAUG.

Basically Acorn have taken the information which was in the User Guide and Advanced User Guide, for the BBC, subtracted those bits not relevant to the Electron, and put the remainder into three smaller books. "Start Programming With the Electron" which does contain a large amount of new material, "The Electron User Guide", and now the "Electron Advanced User Guide".
The first thing which I noticed when the EAUG finally arrived, was its size. It has less than half the number of pages of the BBC AUG. Part of this is because of the comprehensive section on using assembler, contained in the EUG. Another reason is that the Electron does not have as many input/output controls as the BBC.

Judging by its cover, this book will be similar to the BBC AUG. Its name is similar, and two of the authors are the same. One noticable difference is that it is published by Acornsoft, rather than independantly, as is the BBC AUG.

So I dived into the EAUG. The first chapter talks about the Acorn design philosophy of making all it's computers compatable with each other as much as possible. For the person just starting to use the many vectored routines and calls in the Acorn Operating System, this makes very worth while reading.

The second section starts going into what these routines and calls can do. Much of this is a straight reprint of information in the BBC AUG, including one case of details relating to an *FX call (soft key expansion, *FX228 - see this months Electron column) which simply has no effect on the Electron. Some of these details have been changed, notably some which are labeled

"This call is reserved for future expansion". In general though, even where a call in not implemented, some details are given on what that call does on the BBC. On some occasions extra explanation has been given. The EAUG also details a number of calls (*FX22 - *FX24 and others) which are not documented on the BBC AUG.

The next section of the EAUG covers paged ROM software. It is very comprehensive, being over twice as long as the BBC AUG's equivalent section. It is very easy to read, with a number of examples of programs (in assembler) which could be used in paged ROM.

Memory usage and hardware are each covered briefly, in the next two sections. Again, they are easy to read.
I was very dissappointed with the hardware section. Twice the authors have pointed out that the Electron has more expansion potential than the BBC. This is because all the 6502 bus lines are available on the Electron expansion connector. They do not, however, offer any projects suggesting use of this great potential. Very few "serious" books have been written about the Electron, none, as far as I can see, about adding hardware. This is a situation which I find very frustating.

I also noticed that this book, which was published just on a year ago, (Sep '84) contains no information at all on disk or Econet upgrade hardware. It does contain some details on the PLUS 1 expansion.
This book is good. It is very readable, and provides a good introduction to advanced software on the Electron. However, I feel that its use for someone not programming in assembler is limited. There is enough information in the EUG to start writing small assembler programs. If you have difficulty with understanding that, do not expect this book to help. There are other books, either written for the BBC or Electron, which are intended to provide this. I will be buying an Electron Advanced User Guide, mainly to have details of all the OSBYTE (*FX) calls. However I don't consider that it will gain the "essential" label attached to the BBC Advanced User Guide

CLASSIFIED ADS

For Sale: BBC Micro, issue 4 P.C. board (2 years old). 1.2 O.S., DOS, Wordwise and Star Database ROMs installed. Mitsubishi 80 track double sided disc drive (1 year old) utility disc and Disc User Manual. Games and Primary Education software. $2,200 ono.

Phone Evan 330-444 Invercargill.

electron column

steve cosgrove

One thing which I have been asked about time and time again is the question of compatibility between programs for the BBC and those for the Electron. It is a question which you will have to "go on asking" - every program is different, and must be looked at as such. There is some hope though, and here I will present a few ideas.

Last month I mentioned some hardware features, namely the fact that the BBC micro has five separate processors. The bulk of compatibility problems are associated with the screen display. Every computer has a trade-off between the quality of it's graphics output, and the amount of memory it takes to display it.

It is a very involved process to get the screen to display any graphics at all! There is a section of RAM which contains details of what is supposed to be on the screen (the screen memory). Many times each second, the computer must read each of these locations, in turn, and send the appropriate signal out the video wires to the screen. (Many books have been written on how this can be done.) When a line in BASIC like 'PRINT "Hi There"' or 'DRAW 100,200' is encountered, the Machine Operating System (MOS), in conjuction with the BASIC interpreter, will change the contents of the screen memory to produce this display.

MODE 0 displays 640 dots in each line across the screen, and 256 lines down the screen. Each dot can be either white (on) or black (off). A byte in memory can store eight 'flags' (see last month's SAS article), so the number of bytes needed by the screen is 640 * 256 / 8. This comes to (just over) 20,000, or 20K. To display the 4 colours needed by MODE 1, two flags are used for each pixel. But there are only 320 dots per line, so the memory requirement is the same, but picture quality is reduced.

How does this affect compatability?
Firstly we have no MODE 7 (Teletext mode). This mode has very poor graphics (effectively 80 * 75 pixels), but uses only 1K of RAM. MODE 6 takes 8K of RAM. Many programs use the extra 7K, and will not run without some modification. Ideas for modification are given in August Beeblet, "No Room - No - Longer". The Acornsoft PASCAL cartridge promises a "pseudo - mode 7" feature, which I await with interest.
Some machine code games use more than one mode on the screen at a time. This cannot easily be done on the Electron.
Any games which use extremly fast-moving graphics are likely to be beyond the Electron, as often these are created by directly controling the 6845 (screen processor). Less complex graphics, as used in the majority of games, will run without modification, though, of course, more slowly. Some games were actually slowed down to make them a reasonable speed on the BBC. An example is SNAPPER. I am told that if the Electron version is run on a BBC, it is impossible to keep up!

Programs written in BASIC are a different proposition. The most common problem with these seems to be lines used to disable the cursor. 'VDU 23,1,0;0;0;0;' and 'VDU 23,1,1;0;0;0;' will work on both computers, however, there are a number of other ways of performing this operation on the BBC, none of which is Electron compatible. Where MODE 7 graphics are used extensivly, modifications to PRINT statements may be needed to make the screen 'look right', but generally, even without modification, the display is readable.

The vast majority of programs appearing in Beeblet will run on the Electron, and, at the risk of never being allowed to use Anne and Gavin's computer again, I must say that SAS runs OK too!

I have tried a number of BBC ROMs in my Electron, and have had no problems so far, but the inability of Barsons to prduce a reliable ROM board for the PLUS 1 limits what I can try.

The last word for compatability is to try what you are considering. Most programs seem to work without modification.

I have a bit more 'free' time at the moment, and would be happy to consider any letters which club members write to me, C/- the User Group. (Address on the back cover.) I have tons of ideas for the column, but always welcome feedback (good or bad). I would also be happy to try to help with incompatable programs, or general programing enquires. Send me a detailed description of what is supposed to happen, and a printout of the program, and any other details you think may be worthwhile. A cassette or disk copy of the program might help. A stamped addressed envelope will also ensure speedy reply/return.

New products do not appear very often for the Electron, so when one does, it is a shame if it doesn't work, and appears over-priced. That is my opinion of a ROM board which is now available. It will not work reliably without a PLUS 3 installed. (See my column last month on interference.) This is disappointing to me, as I had been loaned one to use with PASCAL until the Electron version comes. Some features work, others don't. I am working on a board of my own, which I hope to be able to sell for under $50 before the end of the year.

Steve Cosgrove

UK EDUCATION - a standard operating system ?

The following article from Computer Weekly 11/7/85 counters the suggestion in the July Acorn User that UK schools might have to standardize on MS-Dos:

"Schools U-turn lets NCC study standards

The (UK) government has backtracked on proposals to urge Microsoft's MS-Dos operating system as a standard for micros in schools.

Now it has commissioned the National Computing Centre (NCC) to study the benefits of a wider range of standards.

Patrick Raymont, who is heading the study, says the report "will look at a wide range of issues and not just operating systems". Standardisation of hardware, software and other factors, such as documentation, will be considered.

The Department of Trade and Industry still wants to find out if standardising on a particular operating system would help UK companies devbelop more software for the educational markets in the UK and overseas.

"The true potential for integrating microcomputers into the curriculum as a teaching tool is only beginning to be explored," says industry under secretary John Butcher.

The government wants to stimulate a strong home market for educational software as a base from which to attack lucrative overseas markets."

MAILBAG

Don A. has drawn my attention to two inaccuracies in the July Beeblet. His Function Key definition should have read:

```
*Key8
z%=65:|N:F.f%=&404TO&468S.4:IF!f%<>0 P.CHR$32CHR$32CHR$z%"% =
";!f%"/ &";~!f%:z%=z%+1:N.EL.z%=z%+1:N.|M|O
```

(Switch on the page mode, then list the values of the resident integers A% - Z% in decimal and hex., finally switch page mode off again. If the value of the integer is zero, the integer is NOT listed.)

Secondly:

Page 33 thanks Don A. (me) for a contribution which was in fact provided by Don Fraser. Since we (Don Austin and Don Fraser) both have the foresight to live in Kupe Place, Tokoroa and own the name "Don" as well as a BBC each, I guess confusion is possible. Just remember that I'm the better-looking one!

While I'm at it, I must challenge the comment made by Odie (?) in the review of Oxford PASCAL where he states that "Pascal ... is far superior to BASIC". To some BASICs perhaps, but not BBC BASIC. Based on the assumption that most BBC users are in it for the fun, I consider BBC BASIC superior to PASCAL because
1. you do not have to be so damned structured in your approach to a program - since BBC BASIC is so flexible, it's more friendly.
2. you do not have muck to around with editors, filers, compilers. If some thing goes wrong in a BASIC program, you just change it. It's not that simple in PASCAL's programming - editing - compiling protocol.
3. it's got strengths, sure, but weaknesses too.
4. more people use BASIC anyway.

PASCAL which, like BASIC, was developed as a learn-to-program language, certainly has a place in the programming world, it certainly forces better structured programming, it does have strengths but that does not make it far superior.

A response to Steve W. (p.24 BRIGHT IDEAS). BBC BASIC is not as inefficient at string storage as some other BASICs. In fact I like the way BBC BASIC handles string storage. To me it makes sense and it's fast! Without going into the wheres and whys, because books such as "The BBC Micro, an Expert Guide" by Mike James do so, let me just repeat the hint which has appeared once before in the BEEBLET.

Don't allow the BBC to be "inefficent". Before using a string, predefine it to its maximum length early in the program with, for example, A$=STRING$(20,"x"). Where the "20" is the maximum length and the "x" is any character. This sets up the string in such a way that no inefficiency will occur as long as the maximum length is not exceeded elsewhere in the program.

Indeed if one wants to be super-efficient:

1. one can deduct 8 from the maximum length whenever the maximum length is 16 characters or more long. This means that if you know the maximum length of A$ will not exceed 16 characters, then predefine it as A$=STRING$(8,"X") since BBC BASIC will still set aside an allowance of 16 characters anyway.
2. only use one character string variables e.g. A$, B$, C$ etc.
3. if you do use more than one character string variables, do not start with the same letter, e.g. add$ and count$ is more efficient than count1$ and count2$.

I include a minor program which illustrates the use of pointers and byteallocations to an example INPUT string, AB$.

Further, for DFS users the RAM areas &900 to &AFF can be used to store strings too. I find this particularly useful for strings which are string specific such as P$ = "Press the SPACE BAR to continue". Instead use $&900 = "Press the SPACE BAR to continue" and then PRINT$&900 instead of PRINTP$.

This has two advantages: it leaves more RAM for your BASIC program and makes such strings available to programs which are CHAINed by the first in the same way that the Resident Integers A% - Z% are. Thus the contents of $&900 remain intact until replaced or the micro is switched off.

BUT &900 to &AFF do have jobs allocated, so check that you make compatible use of this RAM space:
&900 to &9BF is for ENVELOPEs 5 - 16 (most people/programs use ENVELOPEs 1 - 4 only though) or for the RS423 output buffer (again, not used much by
most of us).
&9C0 to &9FF is for the Speech Buffer.

&900 to &9FF is for the Cassette output buffer so watch it!!!
&A00 to &AFF is for the Cassette input buffer so watch it!!!
(also the RS423 input buffer).

Another hint for those who use in their own programs

```
PRINT "Press any key to continue"
REPEAT UNTIL GET
             CLS
```

Try PRINT "Press any key to continue "GET$:CLS It saves typing and RAM and has the same effect. But be warned, if one does not type an alphanumeric character, unwanted effects can occur, e.g. press CTRL and L together and the system PRINTs the GET$ which is CLS!

Finally questions regarding "Elite", BBC disc version. Does anyone know: Are there more than two missions? Are there space cities, dredgers, generation ships to be seen?

****

Neville T, Ch'ch:

In reply to the "majority" of Wellington:
SAS has, in my opinion, earned every page of space it has taken. If you take time to examine the programming techniques, you will find that here is a very interesting series of articles - of interest even to non-adventurers. I look forward to future installments!

****

Steve Cosgrove,
in reply to "the majority"

1 Your pseudonym is well chosen. The majority of complaints which I have seen of ours, and similar magazines, are along the lines of yours. However, the complainents have usually done nothing to help the situation, beyond informing the world of how they feel. I thank you for writing. It is one step better than those who simply vent their feelings by telling the neighbours, and cancelling their subscription. The next question is WHO will write anything? If someone were to buy me a PLUS 3, I would be very happy to write a 40 page magazine about it. However, no-one has, so no review is forthcoming. It is most frustrating too when we are asked, by some, to cut costs, while others ask for "better" content. I could write

reviews on all sorts of things if the User Group could employ me to do so over the summer break. Unfortunately, I don't see that as forthcoming either.

2 The plural of mouse is mice.

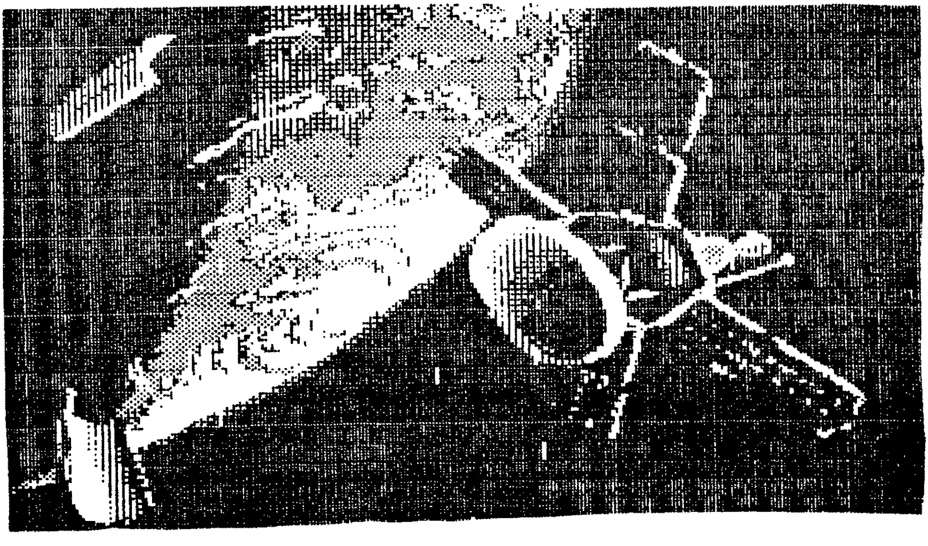
****

Jim 8Logie:

Thats Fantastic

There seems to be an increasing tendency to use a superlative when describing a situation that may require a little imagination to foresee the outcome or propose a solution or just to catch the eye. I recall a program called impossible colours and more recently another program referred to an impossible die. To make a three sided die....When the housework is finished and the wife is not going out for the evening,take a saw and the broom and make two cuts across the handle about 15mm apart and as far from the business end as possible. Now that you have a three sided lump of wood only two problems remain.First numbering it;decimal,hexadecimal,trinary perhaps? And the second is to explain convincingly that your modifications have improved and even enhanced the aerodynamic properties of the broom. The best of incredible luck.

****



BEEBS IN THE BACKWOODS

We never reckon to get many visitors in our neck of the woods, so when I see a dustcloud coming along down McKenzies Track I sit up and take notice. Turns out to be a flashy estate with FARM DATA on the outside and a smarmy type on the inside. Smarmy gets out and "Good afternoon, my man", he says. Well, I don't like that 'cause I'm nobodys' man, so "Good afternoon turnip bonce", I says, only I says it to myself, being more polite than he is.

Seems as if he thinks I need updating because he gets stuck into his spiel about every progressive farmer needing a computer, and not wanting to upset him I says he can give me a demonstration, so in we go and he sets up all his fancy boxes and joins 'em up with bits of wire and then he says we're ready.

I says "What will it do?" and he says "It will answer all your farming questions. Just ask anything you want to know." Well, one thing I would like to know is how Bess, my best bitch (lady dog to you) got in the family way when she was locked in when it must have happened, so I type all that in and the computer says MISTAKE. Well, I know it was a mistake. What I want to know is - how did it happen?

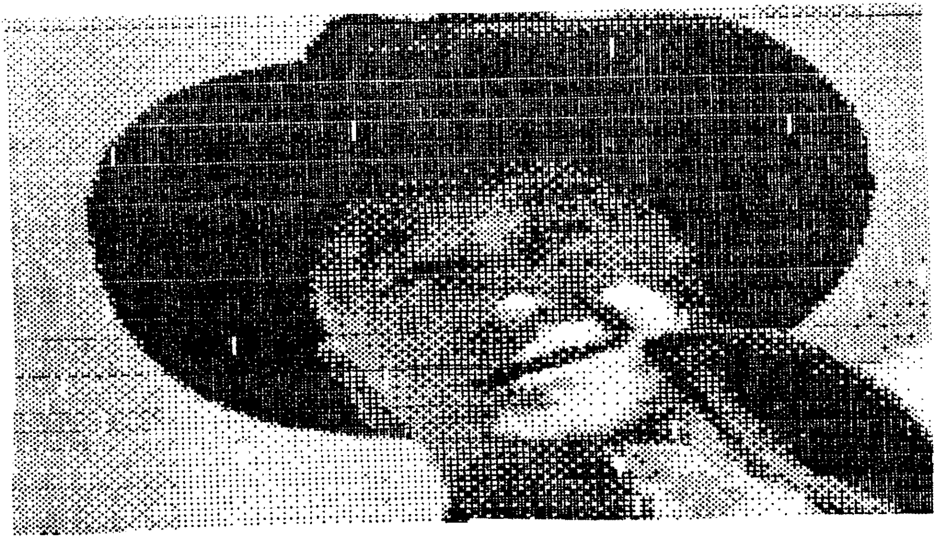
Smarmy smirks and says "No, no, you have to ask it something it's been programmed to answer, like how much wire do you need to fence a paddock, or how much swede seed do you need per hectare. Just type in the length of the sides of the paddock or field and there you are." "Sounds daft to me," I says,"If I know the length of the sides I can work it out for myself."

He isn't too keen on that, so then he asks if I've got anyone to help me with accounts and suchlike and I says yes I have, a girl called Suzie. "Well then", he says, "this computer can do everything she can." I have my doubts about that, but I'll try anything so I put the computer on my knee, but it's not the same at all. When I put Suzie on my knee she puts her arms round my neck and goes all cuddly and - well, never mind.

Old Smarmy by this time has had enough, and gathers up all his boxes, muttering about hayseeds as thick as two short planks having nothing better to do than romp with the dairy maid. Off he goes across the yard, slipping and sliding in one or two country pancakes on the way, and then the dustcloud goes back the way it came, only faster.

Well, he's wrong about me, I'm not thick, because I got a Degree with Honours at Massey in Farm Management, and my column in the Farmers Weekly, "How to survive under a Labour Government", is everyones' favourite. And I wrote the best selling program for dairy farmers, PLOP, i.e. Plan for Longterm Optimum Performance, so I think I know a bit about computers. But I don't need one for working out lengths of fencing wire, it's - just a tick - 2316 metres, or for working out how much swede seed I need, thats - just a tick again - 32.8 kilograms and if you want to know how many there are you can come and count them yourself.

He's wrong about Suzie too, and so are you, because she's my 8 year old daughter, and what she dosn't know about OSBYTE, OPT and the Beeb isn't worth a soft shelled chooky egg. Talking about Suzie and the Beeb reminds me that she should just about have finished her latest two-player, split-screen, interactive battle adventure where the computer throws in random Acts of God to help you on your way. Into the 'den' I go and there she is with 'Battles of Betelgeuse' up and running. "Are you ready Suzie my dear?" I says, and she says "Right Dad", and away we go! YEEOOOWWWwwwww! BOOM!!BOOM!! Pow-pow-pow-pow-pow!!!



SIMPLE DATABASE II

Anne F

Remember my little expose about the Plunket Committee record keeping last time, and how we resolved to alleviate the manual hassles by setting up a DATABASE? Well, here's what we came up with.

DATA STATEMENTS vs DATA FILES

There are advantages in using statements within a program rather than reading in a data file:

1. Not everyone has discs, so a program can't be written to take advantage of some of the disc features which I have (like Random Access which uses less memory than Fixed Length records).
2. Keran has tape. Tape is slow to load in, and even more hassles if you have to find both the program and the data file on tapes.
3. All the information is right in front of your eyes; it is easier then to edit it, fix any errors (eg missing commas!), insert a new person in alphabetic order within all the data.
4. There is only one program with its own self-contained "file". Only one LOAD, only one SAVE, on the occasions when you alter both data and program in the same session (eg add another baby, and perhaps a new PROC if you've thought of a new job to do, such as a list of families with more than 3 under-fives). Only one thing to backup too.
5. Because you have all details in memory, there is a limit on the number of people you can have. Memory space is conserved by putting the list of streets in alphabetic order, and the list of people in alphabetic order of surname. Really? How does that make a difference? By not needing space for PROCsorts in the program, nor the arrays this would require! There is a quick test around line 390 to check the order though.

OTHER USES

Of course, this approach will serve other organisations such as local political party canvassing, boat clubs... Depending on how much information you want to store, you can handle perhaps 300 people or more.

EXPLANATION OF THE PROCs

PROCmenu

Let's make it user friendly, so someone else can use the program.

↓↓

Print up all the options the computer could prepare for you, and prompt for the user's choice.

PROCoutput
is called from all options. It asks whether a printout will be required, in which case the printer will be enabled, or if just a screen browse is wanted, in which case paging mode is set up. Rather than remember which was chosen, both paging and the printer are switched off at the end of each option.

PROCread
This reads through one line of data to define names, street_no, street name, phone, and fills an array with the dates of each child's birthdate.

PROCline
The format used to print out those chosen. Because printers define horizontal TABs in different ways, this facility wasn't used, but TABs are artificially set at W% (20) and 52 characters across by:
calculating the length of names then printing enough extra spaces (X%) to make 20 characters;
expecting address to take up to Y% characters,
making the phone number up to 7 characters so the child ages also start in alignment.

You may care to invent another PROCline for screen output only if you are using a tv, because this one assumes 80 characters across a line. Fine for printers and Mode 3, but if you can't read Mode 3 text and move to Mode 6 or 7 then information won't be so easy to pick out.

PROCstreet_list
This is going to end up with a list in each street, in alphabetical order of street then in surname order. It does this in a perhaps cumbersome manner by taking the first street, reading through all the data and printing out those in that street. It then takes the next street name, and searches through all the data printing out those in that street... until it has gone through the last street, by which time it will have printed out everyone.

The streets are in alphabetical order because that is how they were typed into the DATA statement.

PROCage_test
This is the one to pull out all children in a particular age range. Because there are many ways of entering a date, a sample

→→

format is offered.
It then reads each line of data, checks the array for birthdates in the wanted range and prints out details of any that are wanted.

PROClabel

As talked about last time!

PROCname_list

This produces a list of families in alphabetic surname order.

POTENTIAL PROBLEMS:

1. How many streets have you? What if this number changes?. PROCstreet_array
2. What if the street is spelt one way in the street array, and another within the record of a person? PROCcheck_street.

PROCstreet_array

This restores the line number that starts the list of street names. These have been put at the end of the information about the people. It then reads through them a first time to count how many streets there are, DIMensions an array the right size to hold them all, then reads through them all again to fill that array.
This is only done once in a program.

PROCcheck_array

I've been caught out on occasion by typing errors, or entering someone at "Hohiria St" when it is really "Hohiria Rd" in the array. Of course the computer will ignore this household when it comes to putting out a list in order of street, because it doesn't have "Hohiria St" in its array.

The solution is to check that all the streets can be identified by comparing all street names with those in the array, and printing out those people who don't. If you wish, you can exit the program and any corrections needed.

FUTURE ADDITIONS?

The sky or the computer's memory are the limit.
However, one that might be of use would be a different way of ordering the streets, more in geographic order. This would entail assigning each street a number, and altering the list of street names to include this number. Setting up the array of street names would have to take this into account, and "PROCgeog_list" would look for street number 1, print everyone in that street, find street 2, print all in that street and so on.

```
   10 REM  Plunket database
   20 REM Anne Flower 4/3/85 extended 15/8/85
   30 ON ERROR VDU3,15:REPORT:PRINT" @ line ";ERL:STOP
   40 DIM date(6)
   50
   60 MODE3
   70 PRINT"Setting up and checking the streets..."
   80 PROCstreet_array
   90 PRINT'"In order to do this,"
  100 PRINT"and that surnames are in order"
  110 PROCoutput
  120 PROCcheck_street
  130 REPEAT
  140   CLS
  150   PROCmenu
  160   IF choice$<>"0" THEN PRINT''"FINISHED"'"Press SPACE BAR to return to men
u":REPEAT UNTIL INKEY(-99)
  170   UNTIL choice$="0"
  180 END
  190
  200 DEF PROCstreet_array
  210 RESTORE 1740
  220 count% = 0
  230 REPEAT
  240   READ street$
  250   IF street$ <> "no more" THEN count% = count% + 1
  260   UNTIL street$ = "no more"
  270 DIM name_of_street$(count%)
  280 RESTORE 1740
  290 FOR I%=1 TO count%
  300   READ name_of_street$(I%)
  310   NEXT I%
  320 ENDPROC
  330
  340 DEF PROCcheck_street
  350 previous_name$ = ""
  360 RESTORE
  370 REPEAT
  380   PROCread
  390   IF surname$ < previous_name$ THEN PRINTsurname$;" is out of order" ELSE
previous_name$ = surname$
  400   I%=1
  410   ok%=FALSE
  420   REPEAT
  430     IF street$ = name_of_street$(I%) THEN ok% = TRUE
  440     I%=I%+1
  450     UNTIL ok% OR I%>count%
  460  IF NOT oK% THEN PRINTsurname$;" ";first_name$;" ";street
_no$;" ";street$
  470  UNTIL surname$="END"
  480 PRINT''"Press SPACE BAR to return to menu":REPEAT UNTIL IN
KEY(-99)
  490 ENDPROC
  500
  510 DEF PROCmenu
  520 PRINT'"Enter the number corresponding to"'"the task you wi
sh to do:"
  530 PRINT'"1  list in order of street"
  540 PRINT'"2  list within age range"
  550 PRINT'"3  address labels"
  560 PRINT'"4  list in surname order"
  570 PRINT''"0  Exit program"
  580 PRINT''
  590 REPEAT
  600  PRINT"Your choice...";
  610  INPUT choice$
  620  UNTIL INSTR("12340",choice$)
  630 PRINT'''
  640 IF choice$="1" THEN PROCstreet_list
  650 IF choice$="2" THEN PROCage_test
  660 IF choice$="3" THEN PROClabel
  670 IF choice$="4" THEN PROCsurname_list
  680 ENDPROC
  690
  700 DEF PROCstreet_list
  710 PROCoutput
  720 PRINT"This list is in order of street"''
  730 FOR I% = 1 TO count%
  740  RESTORE
  750  REPEAT
  760   PROCread
  770   IF (street$ = name_of_street$(I%)) AND (surname$<>"END
") THEN PROCline
  780   UNTIL surname$ = "END"
  790  NEXT I%
  800 VDU3,15
  810 ENDPROC
  820
  830 DEF PROCage_test
  840 RESTORE
  850 PRINT'"Please use this format for entering age range:"
  860 PRINT'"yy.mm    -   yy = year eg 1984"
  870 PRINT"          -   mm = month  eg 01 for Jan"
  880 PRINT"so it looks like   84.01"
  890 INPUT'"What is the oldest birth date?  " valid_oldest_age
  900 INPUT'"What is the youngest birth date?  " valid_youngest_
age
  910 PROCoutput
  920 PRINT"List with children born from ";valid_oldest_age'"and
 no later than ";valid_youngest_age''
  930 REPEAT
  940   PROCread
  950   FOR I%=1 TO Kids%
  960      IF date(I%) < valid_youngest_age AND date(I%) > valid_
oldest_age THEN PROCline
  970     NEXT I%
  980   UNTIL surname$="END"
  990 VDU3,15
 1000 ENDPROC
 1010
 1020 DEF PROClabel
 1030 RESTORE
 1040 CLS
 1050 PROCoutput
 1060 REPEAT
 1070   PROCread
 1080   IF surname$<>"END" THEN PROCprint_label
 1090   UNTIL surname$="END"
 1100 VDU3,15
 1110 ENDPROC
 1120
 1130 DEF PROCsurname_list
 1140 RESTORE
 1150 PROCoutput
 1160 PRINT"List in order of surname"''
 1170 REPEAT
 1180   PROCread
 1190   IF surname$<>"END" THEN PROCline 1200   UNTIL surname$="
END"
 1210 ENDPROC
 1220
 1230 DEF PROCprint_label
 1240 PRINTfirst_name$;" ";surname$
 1250 PRINTstreet_no$;" ";street$
 1260 PRINT"Hataitai"
 1270 PRINT"Wellington"
 1280 PRINT'
 1290 ENDPROC
 1300
 1310 DEF PROCoutput
 1320 PRINT''"Do you want screen or printer output?"
 1330 PRINT'"     S for screen"'"or"'"     P for printout"
 1340 INPUTA$
```

```
↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓

 1350 REPEAT UNTIL A$="S" OR A$="P"
 1360 CLS
 1370 IF A$="P" THEN VDU2 ELSE VDU14:PRINT'"Remember to press SH
IFT to move down the screen"''
 1380 ENDPROC
 1390
 1400 DEF PROCread
 1410 Kids%=0
 1420 READ surname$, first_name$, street_no$, street$, phone$
 1430 REPEAT
 1440   Kids%=Kids%+1
 1450   READ dob$
 1460   date(Kids%) = VAL(dob$)
 1470   UNTIL RIGHT$(dob$,1) = "@"
 1480 ENDPROC
 1490
 1500 DEF PROCline
 1510 LOCAL U%,V%,W%,X%,Y%,Z%,I%
 1520 U% = 7: W% = 20: Y% = 32
 1530 X% = W% - (LEN(surname$) + LEN(first_name$))
 1540 Z% = Y% - (LEN(street_no$) +1 + LEN(street$))
 1550 V% = U%-LEN(phone$)
 1560 PRINTsurname$;", ";first_name$;SPCX%;street_no$;" ";street
$;SPCZ%;phone$;SPCV%;
 1570 FOR I%=1 TO Kids%
 1580   PRINT" ";date(I%);
 1590   NEXT I%
 1600 PRINT
 1610 ENDPROC
 1620
 1630 DATA Alfrich,A,23,Moera Rd,881-145,83.10,85.01@
 1640 DATA Breughel,Patricia,34,Arawa Rd,883-210,82.06@
 1650 DATA Howarth,Lyn,107,Windswept Ridge Rd,881-221,82.08,84.0
9@
 1660 DATA Hugrams,Mary,2,Draken Rd,884-122,82.01,83.02,84.05@
 1670 DATA Kitchener,,44,Arawa Rd,882-531,84.11@
 1680 DATA Manson,H,34 Bot flat,Windswept Ridge Rd,881-222,81.12
,83.08@
 1690 DATA Perrisher,Jo,56,High View Tce,882-345,84.12@
 1700 DATA Smith,Helen,68,Moera Rd,882-987,83.06@
 1710 DATA Smith,Margaret,33,The Slipway,885-112,81.11,84.09@
 1720 DATA END,END,0,Arawa Rd,000-000,00.00@
 1730 REM street names next
 1740 DATA Arawa Rd,Draken Rd,High View Tce,Moera Rd,The Slipway
,Windswept Ridge Rd
 1750 DATA no more

↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑
```

BILLBOARD?

Peter C. had sent in a little PROC, plus a demonstration program. He says -

I have called it SCANMES as it scans a message across the screen (within a user defined window) in one of three different ways - 1) appearing at the right, moving into position and stopping, 2) moving away from this position and disappearing on the left, or 3) appearing at one side and disappearing at the other.

The listing includes the procedure plus a demonstration program plus another little PROC that draws a box at the appropriate place on the screen. This last PROC is not necessary but is a little bonus.

It is not, of course, necessary to type in the demo program. One could just type in SCANMES and start using it, but the demo program should demonstrate the various features - and may even include one or two little surprises!

PARAMETERS
A$ = the message
X% = the x position of the beginning of the window
Y% = the y position of the beginning of the window
WLEN% = the window length
FLAG% = flag 1) move in
              2) move out
              3) move straight through

VARIABLES
H% = x position of beginning of message
I% = number of spaces preceding the message
J% = length of the visible part of the message
K% = number of spaces following the message
L% = total length of the message
N% = x position of the beginnning of the visible part of the message
H2% = final location of the beginning of the message (H%)

The speed of the message can be varied in line 670. It can also be sped up by holding down any key.

There may be a quick and simple way of doing this same thing but just in case there isn't, here it is.

```
10 REM Driver program for PROCSCANMES
20 REM P.J.C.  Kaikoura   May 1985
30 MODE 4
40 CLS
50 PROCWINDOW(1,4,38)
60 PROCSCANMES("THIS PROCEDURE SCANS
   MESSAGES ACROSS THE SCREEN WITHIN
   A WINDOW. THE MESSAGE CAN BE LONGE
   R THAN THE WINDOW.",1,4,38,3)
70 PROCSCANMES("MESSAGES CAN MOVE IN
   ...",1,4,38,1)
80 PROCWINDOW(1,8,38)
90 PROCSCANMES("... AND OUT ...",1,8,
   38,2)
100 PROCSCANMES("... AND STRAIGHT THRO
    UGH.",1,8,38,3)
110 PROCWINDOW(1,11,1)
120 PROCSCANMES("THE WINDOW CAN BE ANY
     LENGTH.",1,11,1,3)
130 PROCWINDOW(4,11,20)
140 PROCSCANMES("THAT SAID, THE WINDOW
     CAN BE ANY LENGTH.",4,11,20,3)
150 PROCWINDOW(5,14,30)
160 PROCSCANMES("MOVING IN AND THEN OU
    T ALLOWS US TO WAIT FOR A RESPONSE
    .",5,14,30,3)
170 PROCSCANMES("SHALL WE CARRY ON YET
     ? (Y/N)",5,14,30,1)
180 REPEAT: UNTIL GET$="Y"
190 PROCSCANMES("SHALL WE CARRY ON YET
     ? (Y/N)",5,14,30,2)
200 PROCSCANMES("MOVING IN A MESSAGE T
    HAT IS LONGER THAN THE WINDOW IS P
    RETTY MEANINGLESS AS ONLY PART WIL
    L BE VISIBLE WHEN IT STOPS.",5,14,
    30,1)
210 PROCWINDOW(12,16,16)
220 PROCSCANMES("PRESS ANY KEY",12,16,
    16,1)
230 A$=GET$
240 PROCSCANMES("MOVING IN A MESSAGE T
    HAT IS LONGER THAN THE WINDOW IS P
    RETTY MEANINGLESS AS ONLY PART WIL
    L BE VISIBLE WHEN IT STOPS.",5,14,
    30,2)
250 PROCSCANMES("PRESS ANY KEY",12,16,
    16,2)
260 PRINT TAB(8,18);"TUNNEL          TU
    NNEL";
270 PROCSCANMES("TRAIN",29,18,11,3)
280 PROCSCANMES("TRAIN",14,18,9,3)
290 PROCSCANMES("TRAIN",0,18,8,3)
300 PROCSCANMES("UNDERSTAND IT NOW (Y/
    N) ?",5,14,30,1)
310 REPEAT: A$=GET$: IF A$="N" RUN
320   UNTIL A$="Y"
330 PROCSCANMES("GOOD",5,14,30,2)
340 PROCSCANMES("BYE BYE FOR NOW",1,4,
    38,3)
350 VDU23,240,96,224,97,127,62,28,8,24
360 B$=CHR$240+CHR$32+CHR$240+CHR$32+C
    HR$240+CHR$32+CHR$240+CHR$32+CHR$2
    40+CHR$32+CHR$240
370 PROCSCANMES(B$,1,4,38,3)
380 END
390
400 DEF PROCWINDOW(X%,Y%,WLEN%)
410 REM This PROC is not necessary. It
     is a bonus. N.B. it is MODE depen
    dent.
420 REM It draws a box at the appropri
    ate place. The box is half a chara
    cter bigger than the window in all
     directions.
430 X1%= X%*32-16: X2%=(X%+WLEN%)*32+1
    6: Y1% = (32-Y%)*32+16: Y2%=Y1%-64
440 MOVE X1%,Y1%: DRAW X1%,Y2%: DRAW X
    2%,Y2%: DRAW X2%,Y1%: DRAW X1%,Y1%
450 ENDPROC
460
470 DEF PROCSCANMES(A$,X%,Y%,WLEN%,FLA
    G%)
480 REM PROC to scan messages across t
    he screen. It is not MODE dependen
    t.
490 REM P.J.C.  Kaikoura   May 1985
500 LOCAL H%,I%,J%,K%,L%,N%,H2%
510 L%=LEN(A$): N%=1
530 IF FLAG%=1 H%=X%+WLEN%: H2%=X%+(WL
    EN%/2)-L%/2
540 IF FLAG%=2 H%=X%+WLEN%/2-L%/2: H2%
    =X%-L%
550 IF FLAG%=3 H%=X%+WLEN%: H2%=X%-L%
560 REPEAT
570   H%=H%-1
580   I%=H%-X%: IF H%(X% THEN I%=0
590   J%=X%+WLEN%-H%
600   IF H%(X% THEN J%=WLEN%
610   IF H%+L%(X%+WLEN% THEN J%=H%+L%-
      X%
620   IF J%)L% THEN J%=L%
630   K%=WLEN%-I%-J%
640   IF H%(X% THEN N%=X%-H%+1
650   Z$=STRING$(I%," ")+MID$(A$,N%,J%
      )+STRING$(K%," ")
660   PRINT TAB(X%,Y%);Z$;
670   S=INKEY(15)
680   UNTIL H%=H2%
690 ENDPROC
```

TITLE PAGES IN MODE 7

JL

The Beeb has a row of fKeys which is said to be one of its desirable features. I thought so too when I bought it but I have made scant use of them. This program rectifies this.

I have found composing title pages to be something to be avoided or kept as simple as possible; all those numbers to learn or look up. And its not possible to write it and see it at the same time, or is it?. Some articles in previous Beeblets show that putting some of the codes on screen can be done with fKeys, shift and control, but what about the graphic bits. And so the following program...

The fKeys have been redefined as follows. SHIFT to base 129. Why waste a key on a nil code? CONTROL to base 144. Same again and it allows separated graphics to be on call on f9 (a bit dodgy but worth it). SHIFT + CONTROL to base 160. Pressing a fKey in conjunction with these selects a code number according to the program and gives the rest of the needed codes on fKeys 0 to 5. The other keys on this row will be dealt with later.

Which leaves the ordinary fKeys base 170 and the task of producing the graphics. The mode 7 graphics consist of blocks containing six (large) pixels and the code numbers used to produce them are a sequence as follows:

To a base of 160 add the numbers in Table 1 according to which pixels you want to light up assuming that the six blocks represent the six_pixel graphic character. Thus 160+1+2+4+8+16 = 191, a character with only the bottom right corner not lit; and 160+4+8 =172, only the middle alight. To make use of this the fKeys have been set up so that pressing f1 corresponds to +1,f2 to +2,f3 to +4 etc. So to get the character you want just press the appropriate keys. To keep it all tidy this procedure is selected by pressing f0 (space bar to exit). f7 prints the character (and clears the array used) while copy prints and saves it for re_use. f8 is programmed to print character 255, saves pressing all those keys. The delete key is not able to be used. So with the fKeys and Control and Shift it is now possible to type all the characters you need and see the effect on the screen at once. Corrections and alterations are made by moving to the desired spot and using the appropriate key. The cursor keys have a procedure of their own; press any one to enter and space bar to exit. All things seem to have a drawback of some sort and using the full screen is something I am not able to do. So the bottom line is not available but it can be used to print the odd userfriendly message. The program will after printing the bottom right hand character (39,23) beep and return the cursor to the start of this row(0,23). Shift and f6 to f9 produce codes similar to copy and cursor and to select flash/steady the RETURN Key must be used first,space bar as usual to exit. The COPY Key except in a PROC selects 135.I don't think its a problem. Note also that the fKeys 1 to 9 select 171 to 179 if pressed alone. The Keys all auto repeat and a little care may be needed to avoid retyping but it is hardly a problem. The graphics and cursor routines do not allow the other options,colour changes etc. inside them; exit to use them. Which only leaves a method to save and reload your masterpiece.

An easy method is to save it as a file. Saveone does this. To use it in a program would require it to be loaded as for a file. It would not be easy to write the data out so that someone else could make use of it, say published in Beeblet. It would be more use to have the information saved as a program in numbered VDU statements. Savetwo saves the screen as vdu statements with line numbers and carriage return. *EXEC"yourscreen" with say 1 REM 100 END will give a BASIC program which can be saved,loaded run etc in the normal way. It may contain a lot of empty space (32's) but they could be COPYed out. Or you could write a program to do it. It may need to be separate as this one occupies nearly all the memory space. Screens saveone'd can be loaded back into the program for editing but not savetwo. If the savetwo method is not wanted omit all references to two and the last third of the program. PROC's have been designed to Keep the program in units which can be called in other ways if desired. And I have left three fffree Keys fffor you.

+1	+2
+4	+8
+16	+64

table 1

```
 10 REM ** MODE 7 TITLE PAGES **
 20 REM ** USING SHIFT/CONTROL **
 30 REM ** AND FUNCTION KEYS **
 40 REM **      JL  1985     **
 50 REM fffffffffffffforty
 60 REM ffffffffffffflaming
 70 REM ffffffffffffffunction
 80 REM                    fkeys
 90 MODE 7
100 CLS
110 VDU 31,5,11,133,141,77,79,68,69,32
    ,55,32,84,73,84,76,69,32,80,65,71,
    69,83
120 VDU 31,5,12,133,141,77,79,68,69,32
    ,55,32,84,73,84,76,69,32,80,65,71,
    69,83
130 *FX225,170
140 *FX226,129
150 *FX227,145
160 *FX228,160
170 DIM P%(6)
180 DIM one%(39,23)
190 DIM two%(164,23)
200 PROCsettwo
210 CLS
220 VDU 31,1,24,129,32,32,87,65,73,84,
    73,78,71
230 VDU 30
240 *FX4,1
250
260
270 REPEAT
280   X%=0
290   X%=INKEY(0)
300   IF X%=13 PROCflash:X%=0
310   IF X%>135 AND X%<140 PROCcursor:
      X%=0
320   IF X%=160 VDU 140:X%=0
330   IF X%=161 VDU 141:X%=0
340   IF X%=162 VDU 156:X%=0
350   IF X%=163 VDU 157:X%=0
360   IF X%=164 VDU 158:X%=0
370   IF X%=165 VDU 159:X%=0
380   IF X%=166 PROCsaveone:X%=0
390   IF X%=167 PROCload:X%=0
400   IF X%=168 PROCsavetwo:X%=0
410   IF X%=170 PROCpixel
```

```
420   IF X%=127 X%=0
430   IF X%>0 VDU X%
440   IF VPOS=24 VDU 31,0,23,7
450   UNTIL X%=999
460 END
470
480 DEF PROCflash
490 x%=POS:y%=VPOS
500 VDU 31,1,24,131,136,70,76,65,83,72
    ,130,137,83,84,69,65,68,89
510 VDU 31,x%,y%
520 *FX4,0
530 REPEAT
540   X%=INKEY(0)
550   IF X%=136 VDU 136
560   IF X%=137 VDU 137
570   UNTIL X%=32
580 x%=POS:y%=VPOS
590 VDU 31,1,24,32,32,32,32,32,32,32,3
    2,32,32,32,32,32,32,32
600 VDU 31,1,24,129,32,32;87,65,73,84,
    73,78,71
610 VDU 31,x%,y%
620 *FX4,1
630 ENDPROC
640
650 DEF PROCcursor
660 x%=POS:y%=VPOS
670 VDU 31,1,24,133,32,32,67,85,82,83,
    73,78,71
680 VDU 31,x%,y%
690 REPEAT
700   F%=GET
710   IF F%=136 AND POS>0 VDU 8
720   IF F%=137 AND POS<39 VDU 9
730   IF F%=138 AND VPOS<23 VDU 10
740   IF F%=139 AND VPOS>0 VDU 11
750   UNTIL F%=32
760 x%=POS:y%=VPOS
770 VDU 31,1,24,129,32,32,87,65,73,84,
    73,78,71
780 VDU 31,x%,y%
790 ENDPROC
800
810
820 DEF PROCpixel
830 x%=POS:y%=VPOS
840 VDU 31,1,24,133,32,32,32,80,73,88,
    69,76,83
 850 VDU 31,x%,y%
 860 FOR p=0 TO 6
 870   P%(p) = 0
 880   NEXT
 890 REPEAT
 900   KEY%=GET
 910   IF KEY%=171 P%(1)=1
 920   IF KEY%=172 P%(2)=2
 930   IF KEY%=173 P%(3)=4
 940   IF KEY%=174 P%(4)=8
 950   IF KEY%=175 P%(5)=16
 960   IF KEY%=176 P%(6)=64
 970   IF KEY%=177 PROCprint
 980   IF KEY%=178 VDU 255
 990   IF KEY%=135 PROCrepeat
1000   IF KEY%=9 VDU 9
1010   IF VPOS=24 VDU 31,0,23,7
1020   UNTIL KEY%=32
1030 X%=0
1040 x%=POS:y%=VPOS
1050 VDU 31,1,24,129,32,32,87,65,73,84,
    73,78,71
1060 VDU 31,x%,y%
1070 ENDPROC
1080
1090
1100 DEF PROCprint
1110 PROCsum
1120 FOR zero = 1 TO 6
1130   P%(zero)=0
1140   NEXT
1150 ENDPROC
1160
1170
1180 DEF PROCrepeat
1190 PROCsum
1200 ENDPROC
1210
1220
1230 DEF PROCsum
1240 sum%=0
1250 FOR total=1 TO 6
1260   sum%=sum%+P%(total)
1270   NEXT
1280 sum%=sum%+160
1290 VDU sum%
1300 ENDPROC
```

```
1310
1320
1330 DEF PROCsaveone
1340 PROCstoreone
1350 CLS
1360 VDU 31,0,24,133
1370 INPUT SPC(3)"NAME OF FILE  "F$
1380 F%=OPENOUT(F$)
1390 FOR row = 0 TO 23
1400   FOR column = 0 TO 39
1410     S%=one%(column,row)
1420     BPUT#F%,S%
1430     NEXT column
1440   NEXT row
1450 CLOSE#F%
1460 CLS
1470 VDU 31,1,24,129,32,32,87,65,73,84,
     73,78,71
1480 VDU 30
1490 ENDPROC
1500
1510
1520 DEF PROCstoreone
1530 VDU 31,1,24,133,32,32,32,83,65,86,
     73,78,71
1540 VDU 30
1550 FOR row = 0 TO 23
1560   FOR column = 0 TO 39
1570     look%=(FNscreen(&87,0,0) AND &
         FF00) DIV &100
1580     one%(column,row)=look%
1590     VDU 9
1600     NEXT column
1610   NEXT row
1620 ENDPROC
1630
1640
1650 DEF FNscreen(A%,X%,Y%) = USR&FFF4
1660
1670
1680 DEF PROCload
1690 VDU 31,0,24,133
1700 INPUT TAB(3,24)"NAME OF FILE  "F$
1710 CLS
1720 VDU 31,1,24,133,32,32,76,79,65,68,
     73,78,71
1730 F%=OPENUP(F$)
1740 FOR row = 0 TO 23
```

```
1750   FOR column = 0 TO 39
1760     S%=one%(column,row)
1770     T%=BGET#F%
1780     one%(column,row)=T%
1790     NEXT column
1800   NEXT row
1810 CLOSE#F%
1820 PROCpeep
1840 VDU 30
1850 ENDPROC
1860
1870
1880 DEF PROCpeep
1890 VDU 30
1900 FOR row = 0 TO 23
1910   FOR column = 0 TO 39
1920     VDU one%(column,row)
1930     NEXT column
1940   NEXT row
1950 x%=POS:y%=VPOS
1960 VDU 31,1,24,129,32,32,87,65,73,84,
     73,78,71
1970 VDU 31,x%,y%
1980 ENDPROC
1990
2000
2010 DEF PROCsavetwo
2020 PROCstoretwo
2030 VDU 31,0,24,133
2040 INPUT SPC(5)"NAME OF FILE  "F$
2050 F%=OPENOUT(F$)
2060 FOR row = 0 TO 23
2070   FOR column = 0 TO 164
2080     S%=two%(column,row)
2090     BPUT#F%,S%
2100     NEXT column
2110   NEXT row
2120 CLOSE#F%
2130 CLS
2140 VDU 31,1,24,129,32,32,87,65,73,84,
     73,78,71
2150 VDU 30
2160 ENDPROC
2170
2180
2190 DEF PROCstoretwo
2200 VDU 31,1,24,133,32,32,32,83,65,86,
     73,78,71
```

```
2210 VDU 30
2220 FOR row = 0 TO 23
2230   FOR column = 0 TO 39
2240     look%=(FNscreen(&87,0,0) AND &
         FF00) DIV &100
2250     IF look%>32 PROCencode ELSE PR
         OCspace
2260     VDU 9
2270     NEXT column
2280   NEXT row
2290 ENDPROC
2300
2310
2320 DEF PROCspace
2330 loc = 5+4*column
2340 two%(loc,row)=0
2350 two%((loc+1),row)=51
2360 two%((loc+2),row)=50
2370 ENDPROC
2380
2390
2400 DEF PROCencode
2410 loc = 5+4*column
2420 f%=48+(look% DIV 100)
2430 IF f%=48 f%=0
2440 two%(loc,row)=f%
2450 two%((loc+1),row)=48+((look% DIV 1
     0) MOD 10)
2460 two%((loc+2),row)=48+(look% MOD 10
     )
2470 ENDPROC
2480
2490
2500 DEF PROCsettwo
2510 FOR column = 0 TO 1
2520   FOR row = 0 TO 23
2530     READ list
2540     two%(column,row)=list
2550     NEXT row
2560   NEXT column
2570 FOR row = 0 TO 23
2580   two%(2,row)=86
2590   NEXT row
2600 FOR row = 0 TO 23
2610   two%(3,row)=68
2620   NEXT row
2630 FOR row = 0 TO 23
2640   two%(4,row)=85
```

```
2650   NEXT row
2660 FOR column = 0 TO 160 STEP 4
2670   FOR row = 0 TO 23
2680      two%(column,row)=44
2690     NEXT row
2700   NEXT column
2710 FOR row = 0 TO 23
2720   two%(164,row)=13
2730   NEXT row
2740 ENDPROC
2750
2760
2770 DATA 49,49,49,49,49,49,49,49,49,49
     ,50,50,50,50,50,50,50,50,50,50,51,
     51,51,51
2780 DATA 48,49,50,51,52,53,54,55,56,57
     ,48,49,50,51,52,53,54,55,56,57,48,
     49,50,51
```

	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9
	pixels	▦	▦	▦	▦	▦	▦	print	▦	
SHIFT (text)	red	green	yellow	blue	magenta	cyan	white	flash	steady	
CONTROL (graphics)	red	green	yellow	blue	magenta	cyan	white	conceal	joined	apart
SHIFT + CONTROL	normal	double	black	new	hold	release	save	load	vdu's	

CIVILISED ROUNDING

Hugh P.

Recently I wanted answers rounded off to 2 significant figures rather than to the nearest tens, hundreds or thousands. The exponent format does this admirably but for many purposes 21000 is preferable to 2.1E4. The following function achieves this.

```
 5000 DEF FNnsigfig(N, n%)
 5010 REM  Round N  to  n% significant figures (N >= 0.1, n% 1 to 9)
 5020 LOCAL @%, L$, R$, N$, Nn
 5030 @% = &01010000+n%*&100
 5040 N$ = STR$(N)
 5050 L$ = LEFT$(N$, n%+1): R$ = RIGHT$(N$, 4)
 5060 Nn = VAL(L$)*VAL("1"+R$)
 5070 =Nn
```

A few words of explanation.

@% is the formatting variable.

@% = &B4B3B2B1
B4 = 01 forces STR$ to use format set by @%
B3 = 01 sets exponent format
B2 sets no. of significant figs. See line 5030
B1 normally sets field width but appears to have no effect here.

If 12543 (N) is being rounded to 2 (n%) significant figures L$ will contain '1.3' and R$ 'E4<sp><sp>' (Line 5050)

PASCAL UPDATE

An update to the review of Oxford Pascal. By Odie... again.!!!!

In the Review of Oxford Pascal (July Issue.), I said that it was not compatable with the Second Processor. I have been informed by Barsons that a new version of Oxford Pascal has now been released which works with the Second Processor and is the same price ($179.).

USER GROUP MEETINGS

AUCKLAND: -meets 2nd Wednesday of the month, 7.30pm, Conference Rooms, Auckland Secondary Teachers College, 60 Epsom Ave, Epsom. ph Dave 770-630 x 518(bus) or Kerry 695-355.
-North Shore: meets last Wednesday of the month, 7.30pm, Salvation Army Hall, Glenfield Road, Glenfield ph Chris 419-0543 (wk)

CHRISTCHURCH: -fortnightly workshops, Monday 6.30pm at Hagley High School. ph. Michael 582-267.

DUNEDIN: -first Thursday of the month. Phone Martin McDowall 42-031

HAWKES BAY: -Alternative months Hastings & Napier. Contact Kendall Napier 435-624, Bob Taradale 446-955, Mitch Hastings 778-235.

INVERCARGILL: -Phone Evan 330-444 Invercargill.

PALMERSTON NORTH: -Phone Tom Skinner 83-009(wk)

TAUPO: -meets every second Wednesday @ 7pm. Contact David 84-215 (h), 85-124 (wk).

TAURANGA: -every second Wednesday. ph. Chris & Jane 65-076.

TIMARU: -Phone Lloyd van der Krogt Timaru 61-412.

TOKOROA: -meets first Friday of each month, Tokoroa High School, 8pm. Phone Brian 67-625 Tokoroa.

WAIKATO: -first Monday of the month 7.30pm Waikato Tech Institute. Ph. Alison or John, Morrinsville 6695.

WELLINGTON: -meets last WEDNESDAY of the month, 7.30pm, Royal Society Lecture Theatre, Turnbull St, Thorndon.
-Workshops: second Thursday of month, Wellington College - ring Don 848-235 or Beryl 862-874 or Mike 785-437.

OTHER CENTRES: - let us know and we will publish details here.

THE NEWSLETTER:

CONTRIBUTIONS:-most welcome. Listings should be sent in on tape or disc. (Tape - 2 copies, one at 300 baud please). Please include written explanation of listing; text and graphics only material welcome too.

DEADLINES:- For a particular issue, the last day for material is the last day of the month prior to publication.

ADVERTISING:- Rates are $20 per half page (camera ready), deadline as above.

Copyright (c) BBC/ACORN COMPUTER USER GROUP NZ INC.
Published monthly except January and mailed to financial members.

MEMBERSHIP:

Membership of the users group is on payment of an annual subscription from April to March. For the 1985/6 year it is: $30.00, plus $5.00 for Auckland & Wellington members as a local branch levy, which entitles attendance at meetings as detailed above. Joining or renewing members get all the back issues of the current year - those joining from 1 October may have half the annual sub waived and only receive newsletters from 1 October.

BBC/ACORN COMPUTER USER GROUP NZ INC,
P.O.Box 9592, WELLINGTON.

Reproduced by BRYCE FRANCIS LTD, WELLINGTON.